Richmond Journal of Law and Technology

Volume 1 | Issue 1 Article 7

1995

Overreaching Provisions in Software License Agreements

Michael Liberman
University of Richmond

Richmond Journal of Law &Technology

Overreaching Provisions in Software License Agreements

by Michael Liberman

April 10, 1995

Cite As: Michael Liberman, Comment, *Overreaching Provisions in Software License Agreements*, 1 RICH. J.L. & TECH. 4 (1995) <http://www.richmond.edu/jolt/v1i1/liberman.html>[**].

I. Introduction

{1} Historically, software license agreements emerged as the most popular means of protection of proprietary rights in computer software. As a common form of contract and trade secret protection, software licenses coexist with other forms of intellectual property rights such as patent and copyright. The importance of these forms of protection has recently increased. Where the licensor fails to consider the implications of the relation between these forms of protection, the licensor's attempts to maximize contractual protection while restricting the licensee's activities regarding the licensed software may result in overreaching. Under these circumstances, a court may invalidate the license agreement in whole or in part.

{2} Software license agreements serve several functions in transactions involving the transfer of computer technology. One of the most important legal functions is protection of the proprietary rights of the licensor in the transferred software. Other functions include controlling the revenue streams generated by licensed software and determining the rights and responsibilities of the parties regarding the performance of the licensed technology. Issues related to these functions encompass the applicability of Article 2 of the Uniform Commercial Code, including offer and disclaimer of warranties, determining the appropriate types of licenses to utilize, such as single user/CPU licenses, site/enterprise licenses, and network/concurrent licenses.[1]

{3} Although these issues are important, they are beyond the scope of this paper. The paper focuses primarily on the contractual protection of the intellectual property rights of the licensor, limitations on such protection imposed by federal copyright law and instances where the license agreement may be unenforceable due to overreaching. Since there is no black letter legal definition of what constitutes overreaching, this paper defines overreaching as defeating oneself "by going too far or by doing or trying to gain too much."[2]

{4} Section II explores the role and functions of the software license agreement in the context of the interrelations between patent, copyright, trade secret and contract law.

{5} Section III discusses instances of overreaching and the leading court decisions invalidating software license agreements.

{6} Section IV analyzes the rationale of the court decisions, their practical implications, and ways of

avoiding overreaching provisions in contractual arrangements.[3]

II. Applicability of Different Branches of Law to Software License Agreements

{7} Proprietary rights in computer software can be protected by copyright, patent, trade secret and contract law. Originally, the two most common means of software protection were those available under contract law and trade secret law. Later, copyright law became another prominent form of protection. Only recently has patent protection for computer software become practically available.

{8} Trade secret, copyright and patent law are "static" forms of protection in the sense that they may exist independently of any underlying business transactions and do not necessarily require any transfer of intellectual property from one party to another. In contrast, the need for a license agreement usually arises as one of the contractual forms of protection when the underlying business transaction involves the transfer of intellectual property, such as computer software. In this sense, a software license agreement is a "dynamic" form of intellectual property protection.

{9} Transactions involving the transfer of computer software are subject to both federal and state laws. Generally, state law governs contractual and trade secret aspects of the transaction, while federal law governs aspects related to patent, copyright and antitrust issues.[4]

{10} The United States Supreme Court has addressed the interrelation between federal and state law numerous times.[5] The Supreme Court has consistently held that state laws must yield when they directly or indirectly afford protection that clashes with the objectives of federal patent or copyright law.[6] As the Court stated in *Sears, Roebuck & Co. v. Stiffel Co.*, when state law touches upon the area of patent or copyright statutes, the benefits of the federal policy may not be denied by state law.[7]

{11} The objectives of both patent and copyright laws are based on the power of the Congress "[t]o Promote the Progress of Science and useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries."[8]

{12} A patent gives an inventor "the right to exclude others from making, using, or selling the invention throughout the United States."[9] In order to qualify as a patentable invention, the "process, machine, manufacture, or composition of matter, or ... improvement thereof"[10] must meet rigorous requirements of novelty, utility and non-obviousness, codified in sections 101, 102 and 103 of the Patent Act.[11] Because of the historical difficulty of defining computer software as patentable subject matter, patents were used less often than copyright for the protection of software. However, this has recently changed.

{13} Copyright law applies to "original works of authorship fixed in any tangible medium of expression"[12] and "[i]n no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work."[13] Therefore, copyright protects the expression of an idea but not the idea itself. Copyright protection begins from the moment the work is fixed in a tangible form. According to the Copyright Act, federal copyright law preempts all other "legal or equitable rights that are equivalent to any of the exclusive rights within the general scope of copyright . . . and [that] come within the subject matter of copyright . . . . "[14]

{14} There is no dispute that computer software is subject matter that can be protected under copyright law. After the 1980 amendment, the Copyright Act contains language specifically covering both literal and non-literal elements of computer software.[15] Literal elements include both source and object code.[16] Non-literal elements include user interface and the structure (architecture) of the program. Although courts may differ as to the rationale and sequence of logical steps necessary to extract layers and components of a

program structure that can be protected,[17] the general rule is that non-literal elements of the program can be protected by copyright.

{15} Among the exclusive rights that copyright law grants to a copyright owner are the rights to perform or authorize copying, modifying (preparing derivative works), displaying and distributing of copies of the program.[18] Therefore, if the literal and non-literal elements of computer software can be protected under federal copyright law, the licensor assumes a certain risk that the contractual protection relied upon may be ineffective because of federal law preemption.

{16} As a form of contractual protection, a license agreement may serve several functions. It may establish the necessary relationships between the parties to effectuate other forms of protection, for example, trade secret protection. It may also provide some additional means of protection under state common law where either federal or state statutory protection is unavailable.

{17} The Uniform Trade Secrets Act codifies the common law of trade secrets[19] and has been adopted in some form by a majority of states.[20] "Trade secret" means

> information, including ... a formula, pattern, compilation, program, device, method, technique, or process, that: (1) derives independent economic value, actual or potential, from not being generally known to, and not being readily ascertainable by proper means by, other persons who can obtain economic value from its disclosure or use, and (2) is the subject of efforts that are reasonable under the circumstances to maintain its secrecy.[21]

{18} The factors to be considered in deciding whether information constitutes a trade secret are set forth in the Restatement of Torts: (1) the extent to which the information is known outside of the business; (2) the extent to which it is known by employees involved in the business; (3) the extent of measures taken by the owner of the business to guard the secrecy of information; (4) the value of the information to the owner of the business and to his competitors; (5) the amount of efforts expended by the owner of the business in developing the information; and (6) the amount of efforts which should be expended in order to properly acquire or duplicate the information by others.[22]

{19} Unlike a patented invention, which is disclosed to the public, the innovation or software to be protected as a trade secret should be reasonably kept in secret from the public. Another distinction is that "unlike the standard applicable to an applicant for a patent, the trade secret proponent is not required to prove its claim nationwide."[23] A license agreement may be a part of reasonable efforts to maintain secrecy of the information and an essential part of securing trade secret protection. The license agreement can accomplish this by contractually establishing confidential relations between the parties and restricting the dissemination and use of the information outside of the business of the information owner.

{20} Where necessary elements of trade secret protection are lacking, a license agreement may create rights and obligations between the parties which are intended to protect, by contract, the intellectual property rights of the licensor.[24] For example, although the standard of novelty for a trade secret is substantially lower than for a patent, there is still a requirement that trade secret information possess at least a minimal degree of novelty.[25] Where the software at issue is lacking a necessary degree of novelty, courts have denied trade secret protection.[26] However, even under such circumstances a license agreement may protect licensed software as valuable know-how. A program may be valuable merely because it already exists and performs some useful function, and the party who acquires the right to use such a program saves considerable expense on development. In this situation contractual common law protection created by a license agreement may be the only protection available for the licensor, and therefore the only legal means to support the underlying business transaction.

{21} Another benefit of contractual protection in the form of license agreements is the ability to strengthen the licensor's position in asserting rights under other forms of intellectual property protection. For example, including notification of the licensor's copyright in a license agreement may make it easier for the licensor (in case of infringement by the licensee) to recover either damages for the infringement,[27] or an award of statutory damages. As for trade secrets, license agreement provisions for confidentiality, reasonable non-competition and reasonable restriction on the use and distribution of licensed software may facilitate the licensor's ability to prevent competitors from obtaining licensed programs. This ability is not automatically provided by trade secret law.

{22} A license agreement, as opposed to a sale, allows the licensor to retain the title to the software which goes into possession of the licensee under the agreement. This is essential for establishing contractual obligations facilitating both trade secret and copyright protection of the licensor.

{23} After enactment of the Computer Software Rental Amendments Act in 1990[28] federal copyright law prohibits licensees *and* purchasers from leasing the computer programs without permission of the copyright owner.[29] There are still significant differences in the rights of purchasers and licensees. Perhaps the most significant of these differences is based on the first sale doctrine, under which a purchaser (as opposed to a licensee) of a copyrighted item has the right to sell that copy or dispose of it in any way without the permission of a copyright owner.[30] By contracting for a license agreement as opposed to a sale, the owner of the intellectual property retains the right to exercise control over further disposition of the software. The licensor can also attempt to contractually restrict certain activities regarding the licensed software.

{24} There are certain limitations on what can be accomplished by contractual means where issues of public policy are involved,[31] especially relating to restraints on competition.[32] "The common law courts would require a substantial demonstration of the need for a restraint. Courts value highly the right to compete free from contractual restraint and have permitted restraints no broader than those required by the justification." [33]

{25} Another limitation may be imposed by the relevant state statute -- "a contract cannot be used to go beyond the limits of the statutory right. Under our law, as it has developed, the statutes provide the limits to contract."[34] "This tradition suggests that courts should not expand the scope of intellectual property rights simply because owners assert such rights and claim a loss of value if the rights are not recognized."[35] *Vault Corp. v. Quaid Software* confirmed the theory that in addition to private contractual arrangements, state statutes are without power to extend the licensor's protection beyond the limits of the statutory rights established by federal law.[36]

III. Leading Court Decisions Invalidating Software License Agreements

{26} Adequate employment of software license provisions gives the licensor an opportunity to effectively protect its intellectual property rights in licensed software. However, in some instances the licensor is overzealous in asserting its rights and stretching them beyond the scope of state trade secret and contract protection, "trespassing" into the area of federal copyright or patent law. Ordinarily this occurs when the licensor tries to use as much protective language as possible without proper analysis of the licensed software and the interrelation of state and federal law. According to one commentator,

> once lawyers persuaded software developers that they could not sell their programs like books and instead had to demand that their customers sign onerous license agreements as a condition to access to the software, the floodgates were opened for lawyers to pile into the agreements all protection they could think of for their clients.[37]

In such cases, the licensor may be exposed to the alleged infringer's challenge, based generally on the grounds of federal copyright law preemption. The court decisions discussed in this section demonstrate that such a challenge may be successful even where the licensee in fact breaches the license agreement or infringes the copyright. It has been noted that "in the area of activity restrictions . . . the courts and legislatures have been somewhat active, both in setting aside restrictions they perceive as overreaching, and in providing new law to deal with legitimate concerns that vendors have had to address through their licenses."[38]

{27} The facts in two of the discussed cases - *Sega Enterprises v. Accolade, Inc.*[39] and *Atari Games Corp. v. Nintendo of America*[40] - do not directly involve software licenses. In these cases, both defendants purchased and reverse-engineered software products without the permission of the developer and copyright owner. However, these decisions are discussed here because the prohibition on copying and reverse engineering of the software is a common provision for software licenses where the software is licensed in the form of object code. The rationale of the court decisions in Sega and Atari is applicable to software license disputes and can serve as a guideline for the drafter of a license agreement attempting to avoid overreaching.

{28} Other decisions discussed in this section deal with software license agreements of different types, ranging from the box-top (or shrink-wrap)[41] licenses in *Vault Corp. v. Quaid Software*[42] to customized software in *Lasercomb America v. Reynolds*.[43]

{29} In *Vault Corp. v. Quaid Software*, the plaintiff developed a program to protect a certain type of magnetic disk from unauthorized copying. The defendant copied plaintiff's program, analyzed how it worked and based on this analysis, developed its own program. The main purpose of this program was to defeat Vault's protection scheme. In addition, the second version of Quaid's software contained approximately thirty characters copied from Vault's program. The shrink-wrap license agreement accompanying Vault's software specifically prohibited the licensee from transferring, sublicensing, renting, leasing, conveying, copying, modifying, translating, converting to another programming language, decompiling or disassembling[44] the licensed software for any purpose without the licensor's prior written consent.[45]

{30} Vault brought an action seeking an injunction and monetary damages. The plaintiff asserted that Quaid committed copyright infringement by (1) copying Vault's program for the purpose of developing a program designed to defeat the function of Vault's program, and (2) copying approximately thirty characters of Vault's program into Quaid's program, which Vault contended constituted a derivative work.[46]

{31} Vault also asserted two claims based on state law. First, it contended that Quaid breached the license agreement by decompiling and disassembling Vault's program in violation of the Louisiana Software License Enforcement Act[47] ("Louisiana's License Act"). Second,Vault claimed that Quaid misappropriated Vault's program in violation of the Louisiana Uniform Trade Secrets Act[48] (this latter claim was abandoned by Vault on appeal).

{32} The Fifth Circuit relied on the Copyright Act, which specifically states that

> it is not an infringement for the owner of a copy of a computer program to make or authorize the making of another copy or adaptation of that computer program [when such new copy or adaptation] (1) ... is created as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner, or (2) ... is for archival purposes only and that all archival copies are destroyed in the event that continued possession of the computer program should cease to be rightful.[49]

{33} The court refused to follow Vault's interpretation of § 117 under which an exception of § 117(1) would permit the copying of a computer program only for the purpose of using it as intended by the copyright

owner. Therefore, the court decided that "Quaid did not infringe Vault's exclusive right to reproduce its program in copies under § 106(1)."[50]

{34} The court then addressed the issue of the derivative work. The Copyright Act provides that the copyright owner has an exclusive right to prepare derivative works of the copyrighted source material.[51] An infringing derivative work must contain in some form a portion of the copyrighted work and both works must be substantially similar.[52] In the case at bar, eighty pages of Quaid's work contained thirty characters from Vault's program. Vault contended that this sequence of characters was essential to the operation of its program and was also crucial to the ability of Quaid's software to defeat the protective function of Vault's program; therefore, as Vault argued, the copying was qualitatively significant.[53] Vault relied heavily on the *Whelan Associates* decision, which states that a "court must make a qualitative, not quantitative, judgment about the character of the work as a whole and the importance of the substantially similar portions of the work."[54]

{35} The *Vault* court distinguished *Whelan Assocs*. based on the fact that in *Whelan Assocs*. the derivative work performed essentially the same function as the copyrighted work, whereas in *Vault*, Quaid's and Vault's programs served opposing functions, and were not substantially similar. Therefore, the court concluded that Quaid's program did not constitute a derivative work of Vault's program.[55]

{36} Addressing Vault's claims under Louisiana state law, the court stated that Louisiana's License Act permitted the inclusion of such terms as the prohibition of (1) any copying of the program for any purpose, and (2) modifying and/or adapting the program in any way, including adaptation by reverse-engineering, decompilation or disassembly in the license agreement.[56] The Fifth Circuit agreed with the district court that there were numerous conflicts between the Louisiana License Act and the Copyright Act.[57] The district court held that Vault's shrink-wrap license agreement was a contract of adhesion and as such it could be enforceable only if the Louisiana License Act was valid and enforceable itself.[58] Citing the Supreme Court's decision in *Sears, Roebuck*, the Fifth Circuit court decided that Louisiana's License Act clearly touched upon an area of federal copyright law and was therefore preempted by federal law.[59] The court concluded that the provision in Vault's license agreement, which prohibited the decompilation or disassembly of its program, was unenforceable under federal law.[60]

{37} In *Lasercomb America v. Reynolds*,[61] plaintiff developed a die-making software program called Interact and licensed the object code to Holiday Steel Rule Company ("Holiday"). The license agreement contained non-compete provisions which prohibited a licensee from writing, developing, producing or selling such software during the term of the license agreement (ninety-nine years in that particular case).[62] The licensee was further prohibited from writing, developing, producing or selling or assisting others in the writing, developing, producing or selling computer assisted die making software (for the term of the license agreement and for one year after its termination), directly or indirectly, without Lasercomb's prior written consent.[63] Nevertheless, Holiday copied the program in violation of the license terms, and its programmer, Reynolds, reverse engineered and then slightly modified Lasercomb's program.[64] Holiday then sold it as a Holiday software product, PDS-100. Reynolds developed a function that encrypted the Holiday program's output in order to hide the origin of the program from the customers.

{38} Lasercomb brought an action for copyright infringement, breach of contract, fraud, misappropriation of trade secret, unfair competition and false designation of origin.[65] The district court found for the plaintiff on all counts.[66] One of the defendant's contentions on appeal was that it was error for the district court to reject their copyright misuse defense.[67]

{39} On appeal, the Fourth Circuit decided that the non-compete clause of the license agreement was anti-competitive and contrary to public policy.[68] The court upheld the defendant's copyright misuse defense and held that the plaintiff's copyright, as well as the license agreement, were unenforceable.[69]

{40} The court stated that "[t]he question is not whether the copyright is being used in a manner violative of antitrust law (such as whether the licensing agreement is 'reasonable'), but whether the copyright is being used in a manner violative of the public policy embodied in the grant of a copyright."[70] Although the defendants did not sign the license, they were able to prove that at least one other licensee had signed Lasercomb's standard license agreement which contained an anticompetitive clause. This was sufficient for the court to find that copyright misuse was a valid defense for the defendant. The only count on which the court held for the plaintiff was fraud.[71]

{41} In *Sega Enterprises v. Accolade, Inc.*,[72] the plaintiff, a Japanese company, brought an action for trademark and copyright infringement and unfair competition. Sega manufactured the Genesis console which was used to play video games. The object code which manipulated the console was stored in read-only memory (ROM) which was commercially available. The defendant, an American company in the business of developing computer software, reverse engineered Sega's program from the object code, analyzed it and developed its own software for Sega's console using the information acquired by reverse engineering.[73]

{42} The Ninth Circuit held that the reverse-engineering performed by Accolade satisfied the requirements of fair use.[74] In reaching this decision, the court applied the four factor test for a fair use defense under copyright law: (1) the purpose and character of the use, including whether such use is of a commercial nature or is for nonprofit educational purposes; (2) the nature of the copyrighted work; (3) the amount and substantiality of the portion used in relation to the copyrighted work as a whole; and (4) the effect of the use upon the potential market or value of the copyrighted work.[75]

{43} As for the first factor, the direct purpose of Accolade's copying was to study Sega's program in order to be able to provide the compatibility with Sega's console[76] (although the ultimate goal was to develop its own software for Genesis). In the court's view, this was enough for Accolade to be able to assert a fair use defense.[77]

{44} Analyzing the second factor, the court examined the idea/expression dichotomy. Copyright law protects an expression of an idea, but not the idea itself.[78] Therefore, in the court's view, copyright law does not protect functional elements of computer programs necessary for the operation of a program, because they are not part of a protected expression.[79] Otherwise, a copyright owner would obtain patent-like protection without meeting the rigorous requirements for patentability. The court decided that Sega's software could be afforded a lesser degree of protection than more traditional literary works because it contained unprotected aspects that could not be analyzed without copying.[80]

{45} With regard to the third factor, the court found that Accolade had copied the entire program.[81] This factor favored the plaintiff. However, the court concluded that the ultimate use of the copied program was insubstantial and therefore the third factor was of little weight.[82]

{46} As to the fourth factor, the court held in favor of the defendant. The court concluded that the effect of Accolade's activity upon the potential market was not negative because it would increase the number of programs offered for Genesis consumers.[83] The court found that there was no reason to assume that there would be substantial harm to the market for Sega's products.[84] Even if there were some insignificant harm to Sega's market, in the court's view that was not a sufficient reason for denying Accolade's fair use defense.[85]

{47} The court concluded, therefore, that three of the four factors were in the defendant's favor and that Accolade's copying and reverse-engineering were protected by the fair use defense. In the court's view, reverse-engineering (along with the copying of the original program) is permitted under copyright law where there are no other ways to gain access to the program and understand functional concepts in order to develop compatible products.[86]

{48} In *Atari Games Corp. v. Nintendo of America*,[87] Atari reverse-engineered a computer program developed by Nintendo to prevent the use of unauthorized cartridges on the Nintendo Entertainment System.[88] In order to obtain sufficient information about Nintendo's system, Atari's former counsel obtained source code for the security program developed by Nintendo from the United States Copyright Office under false pretenses. This code was later used in the process of reverse engineering.[89]

{49} Nintendo alleged that Atari had infringed on its copyright by copying the source code of the program obtained from the Copyright Office, copying the program during the reverse engineering process, and creating programs substantially similar to Nintendo's program. Nintendo moved for a preliminary injunction which was granted by the district court.[90]

{50} The Court of Appeals for the Federal Circuit (CAFC) upheld the injunction but disagreed with the district court's rationale as to the issue of reverse engineering.[91] The CAFC held that reverse engineering itself, untainted by the purloined copy of Nintendo's program, was a fair use because it was the only way to examine the object code on Nintendo's chip.[92] The court stated that intermediate copying can be justified as a fair use when it is required by the nature of the work in order to understand the ideas and processes in a copyrighted work.[93] However, the CAFC concluded that, because of its use of a misappropriated copy of Nintendo's source code, Atari should be denied protection under the fair use defense.[94]

IV. Analysis of Court Decisions

{51} There is no clear-cut rule that proscribes a way to avoid overreaching. However, an analysis of leading court decisions will provide some understanding of the grounds on which software license provisions may be invalidated.

A. Lessons to Learn

{52} 1. State trade secret and contract law are preempted by federal copyright law. This is perhaps the most important premise for the court decisions that affect the resolution of all other issues relevant to software license disputes. Whether the legal or equitable right asserted or prohibited by the license agreement in question is equal to any of the exclusive rights within the general scope of federal copyright law is key to deciding the question of the validity of the license agreement.

{53} To resolve this issue, the courts appear to differ in the application of the "equivalence of rights" test. The *Vault*[95] court, in invalidating the license agreement and Louisiana's License Act, found that the provisions of state law clearly conflicted with federal law but did not directly address the "additional element" test. In *Foresight Resources Corp. v. Pfortmiller*,[96] the fifth circuit reached a similar decision, emphasizing that the plaintiff's claim was based "precisely upon the same facts as those underlying plaintiff's copyright infringement claims."[97] The court in *Computer Assocs. v. Altai*[98] held that state law is not preempted where the state claim contains additional elements (for example, a breach of a confidential relationship) which change the nature of the cause of action. In other words, under the "additional element" test, if the elements of the plaintiff's claim under state law are exactly the same as the elements of the claim under federal law, state law is preempted.[99]

{54} As the court decisions suggest, in order to successfully challenge the validity of a license agreement, the licensee may rely on the doctrine of federal law preemption in at least three different ways.

{55} Under the first rationale, if state law affords the licensee fewer rights in the licensed program than

federal copyright law, the license may be held unenforceable.[100] The second option for the licensee is to raise an equitable defense of copyright misuse.[101] The third option is to assert a fair use defense under federal copyright law.[102]

{56} 2. Unreasonable, anticompetitive non-compete clauses may be used by the infringing licensee as an equitable copyright misuse defense.[103] The whole license may be held unenforceable even where the infringing acts are totally unrelated to the non-compete clause itself.[104] The obvious lesson is to draft non-competition provisions more carefully and to avoid anticompetitive language in the license of the copyrighted work. In addition to using a severability clause, the benefits that a particular provision can give must be carefully balanced against the dangerous possibility that if anti-competitive, the whole license agreement becomes invalid.

{57} 3. The enforceability of box-top (shrink-wrap) licenses is highly questionable. The *Vault* court unequivocally treated a license of such type as a contract of adhesion and, therefore, unenforceable.[105] Although a more recent decision in *Step-Saver Data Systems v. Wyse Technology*[106] does not directly address the issue of overreaching, it provides an important insight into the rationale that may be applied to the cases of overreaching. Addressing the disclaimer of warranties made in the box-top license, the *Step-Saver* court stated that where the disclaimers are made available to the potential licensee only after the contract is formed, section 2-207 of the U.C.C., and not the license itself, governs the interpretation of the agreement.[107] Therefore, in the absence of an affirmative act on the part of a prospective licensee indicating that he or she was aware of the terms of the box-top license agreement (for example, where a customer in the computer store fails to read a box-top license and the licensor attempts to enforce such a license), there is a significant probability that the court would find overreaching and hold the license agreement unenforceable in part or in whole.

{58} 4. Courts have held that reverse engineering and disassembly constitute fair use under certain circumstances. As the court held in *Sega*, "we conclude that where disassembly is the only way to gain access to the ideas and functional elements embodied in a copyrighted computer program and where there is a legitimate reason for seeking such access, disassembly is a fair use of the copyrighted work, as a matter of law."[108] Although the *Sega* and *Atari* courts did not deal with this issue in the context of a license agreement, this rationale can be applicable to software license disputes based on the federal law preemption doctrine. A similar rationale was employed by the *Vault* court in invalidating a software license under Louisiana state law.[109]

{59} 5. When applying the fair use defense to the copying and reverse engineering of the software products, courts do not necessarily engage in the analysis aimed at the precise separation of the protected and unprotected elements of the programs (similar to the approaches of the *Lotus*,[110] *Computer Assocs*.[111] or *Gates Rubber*[112] courts) in order to allow copying and reverse-engineering only for the unprotected elements of the program (ideas and facts) and disallow it as for the protected elements (expressions of the ideas). The courts do not reason that the fair use defense may be applicable only to the protected elements of the program because the doctrine of fair use itself is aimed at the justification in some instances of using the protected elements in a manner which, without the fair use defense, would constitute infringement. Instead, it was enough for the*Sega* court to find that allegedly infringing programs contained some unprotected elements under copyright law in order to reject plaintiffs' claims for unpermitted copying and reverse engineering of the whole program.

{60} 6. Judicial interpretation of § 117 of the Copyright Act, as evidenced by the *Vault* opinion, does not allow the licensor to control the manner in which the program may be lawfully used for the purposes of § 117(1)'s exception to the Copyright Act.[113]

{61} 7. Even where the court is required to make a qualitative judgment about the importance of the part of

the program copied by an alleged infringer, the court has the discretion to determine its own criteria for making such a judgment. Where policy considerations are involved which favor promoting competition on the market, the plaintiff's argument that the copied part of the program constitutes the main value of the program and is crucial for its functioning, as well as the functioning of the allegedly infringing program, may be rejected, as it was rejected by the *Vault* court.[114]

B. How to Avoid Overreaching

{62} There are a number of tools of contractual protection available to the parties to a license agreement.[115] A license agreement ordinarily contains a provision where the parties acknowledge the status of the licensed program and licensed documentation as confidential/proprietary information. Confidential/proprietary information (subject to some additional requirements) is information disclosed to the licensee by the licensor or some other authorized party. In addition to the confidential information commonly used in other types of trade secret/know-how license agreements, software license agreements allow the licensor to designate research programs, computer software, program files, source, object and executable codes as confidential.

{63} The *Model Software License Provisions* offer a number of clauses which may be used for the protection of the licensor's proprietary rights and restriction of certain licensee's activities. These clauses determine: (a) security conditions,[116] (b) non-use obligations,[117] (c) non-disclosure obligations,[118] (d) restrictions on unauthorized copying,[119] (e) disclosure procedures ordered by governmental bodies,[120] (f) requirements of no removal of proprietary legends,[121] (g) requirements of reports of third-party misappropriation,[122] and (h) post-termination procedures.[123]

{64} If the licensor wants to minimize the probability of overreaching while restricting the licensee's activity regarding the licensed software, it may consider separating different means of contractual protection. The licensor may then apply only those contractual clauses or devices which are more appropriate to the particular software product or its component (in addition to the means of protection available under federal copyright and patent law).

{65} A good starting point may be a detailed and precise definition of the licensed software. The particular provisions depend on the complexity of the software product. In some cases, a satisfactory definition may require reference to a special supplement or appendix, which may also include a detailed description of important software characteristics.[124]

{66} The licensor may choose to protect the whole software product by taking advantage of the appropriate confidentiality provisions. Such provisions could include, among others, a security conditions clause, a non-disclosure obligation clause and a non-use obligation clause.[125]

{67} It may be beneficial for the licensor to apply different means of legal protection to both the software components which provide interface (compatibility) with hardware and other "external" software components of the system (the "ideas and facts" or "functional elements") and those parts of the program which constitute "the expression of the ideas."

{68} As for the first group, the analysis of the *Sega* and *Atari* decisions leads to an argument for the licensor to not cover such software components with the clause restricting unauthorized copying, duplicating, reverse engineering, reverse compiling and disassembling as long as the non-disclosure obligation clause and other relevant clauses adequately protect the licensor's proprietary rights in these components.

{69} As for the second group of software components, the use of the clause restricting unauthorized copying, duplicating, reverse engineering, reverse compiling and disassembling may be helpful in supplementing and strengthening copyright and trade secret protection.[126] However, while drafting such provisions, the licensor has to take into consideration the limitations imposed on the licensor's exclusive rights conferred by federal copyright law. Otherwise, a licensor's attempt to prohibit certain uses of the software,[127] or to impose stricter limitations on the licensee's right to make copies or adaptations of the copyrighted program in instances specifically permitted by § 117,[128] may result in overreaching. The court decisions in *Vault* and *Lasercomb* support these propositions.[129]

{70} The licensor may also identify crucial components and treat them as separate parts of the licensed software system, specifically prohibiting their copying and reverse engineering. This may help the licensor to avoid a situation similar to that in *Vault*, where the defendant Quaid copied and used the most important and valuable part of the plaintiff Vault's program. The Vault court did not find infringement on the part of Quaid, notwithstanding Vault's arguments that the copied part was essential to its functioning and that the *Whelan Associates* decision directed courts to make a qualitative, not a quantitative, judgment about the importance of the copied part.[130]

{71} The above approach may allow the licensor to adequately protect different functional components of the program with different means of legal protection and to avoid the invalidation of the license agreement because of overreaching. Even if the licensee reverse engineers the software component which is not covered by the clause prohibiting unauthorized copying and reverse engineering, it may not be able to exploit this information under appropriately drafted non-use obligation and non-disclosure obligation clauses. At the same time, the license agreement itself will not be vulnerable to the licensee's allegations of overreaching.

{72} The proposed approach may be illustrated by the following example. Suppose A company develops for B company a communication system consisting of hardware and software components. Hardware is delivered under the contract of sale, and software is licensed out from A to B.

{73} The licensee, B, develops its own application programs which run on different computers connected to the network. In order to interact with each other and to exchange the application data necessary for their work, the application programs use the licensed communication software. Logically these application programs are placed on top of the underlying licensed communication software, which in turn is placed on top of the underlying hardware and communication devices.

{74} The communication software consists of different parts. These parts may be viewed as several layers, functioning on different logical levels, where programs at each level are responsible for the realization of the functions designated to that particular level only. The programs interact with their counterparts on different computers. In order to communicate with each other, different programs at the same level of hierarchy follow special rules of interaction, which are called protocols. The programs of any level may also interact with other software (or, in case of the lowest level - hardware) components residing at the adjacent levels of the vertical hierarchy. The rules governing such interactions are called interfaces.

{75} Application data, which is being transmitted from an application program working on one computer to an application program working on another, is delivered from the highest logical level of software system to the lowest, being processed respectively by the programs of each logical level. Eventually this data is delivered from the programs of the lowest logical level to the hardware components of the system and transmitted through the communication devices and communication channels to the destination computer. There the application data makes its way up through all logical levels of software components and is eventually delivered to an application program-addressee.

{76} For the typical software system described in this example it is common for the licensor to disclose to the

licensee the interface between the programs of the highest logical level and the application programs presumably developed by the licensee. The licensed software is rendered useless to the licensee without this information. However, the interfaces between the program components of different levels (including the interface between the lowest-level software components and the hardware) and the protocols may be proprietary information of the licensor and thus subject to protection under the license agreement.

{77} Licensed software's primary value may be enjoyed in two ways. First, the software components of the lowest level in the logical hierarchy are designed specifically to facilitate the operation of the underlying hardware and communications devices. Thus it is possible for the licensee to acquire the use of the whole system. This information about the interface with the hardware is crucial for writing software intelligently on the lowest level. Otherwise, the whole system hierarchy, including the programs of the higher levels, becomes useless. Thus, the party obtaining such information acquires the principal ability to develop its own software and to use underlying hardware and communication equipment. Such information may be treated as a "fact," "idea" or "functional element." Access to this information makes it possible to exploit the whole system, including both hardware and software components, without regard to the effectiveness of its particular implementation. Technically, the interface between the software components at the lowest level and hardware may be realized through a program. Where there are only one or few possible ways of interacting through the use of predetermined codes, the "expression" of the idea (as implemented in the program) and the "idea" itself (the concrete codes, or bit combinations, of the interface) merge into one and may be treated as an "idea," or "functional element" (in terms of the *Sega* court). As such, it is considered unprotected under copyright law.

{78} The second aspect relates to the performance characteristics of the whole structure of the software system, including all hierarchy levels. Designing, coding, testing, system tuning, eliminating mistakes and other stages of the software life cycle may be much more expensive than hardware components. The logical structures, algorithms and programming techniques employed for the realization of the software may be the primary asset for the party who desires to develop a whole system. The ready and effective software system provides an advantage for the party who gets access to such a system. Therefore, the main value is based upon the practical realization of the programs as a particular expression of scientific and engineering ideas which underlie the design of the software system.

{79} The licensor will most likely seek to protect his intellectual property rights in both aspects of the licensed software: in operations/development facilitation and in performance characteristics. Keeping the information concerning the interface with the hardware secret will prevent unauthorized development of programs which exploit the hardware, the communications equipment and the network. Protecting information about the design, structure and practical implementation of the whole software system will support the licensor's software market position and prevent competitors from gaining an unfair advantage over the licensor.

{80} While it may be tempting for the licensor to invoke as much protective language as possible without regard to the function and structure of the software, the possibility of overreaching should be noted. In addition to the security conditions clause, non-use obligations clause and non-disclosure obligations clause, the licensor may include provisions prohibiting the licensee from copying, duplicating, reverse-engineering, reverse-compiling and disassembling of any part of the licensed software system. Later, if the licensee copies and reverse engineers the licensed software, including the components implementing interaction with the hardware and communication equipment, the licensor may bring an action for breach of the license agreement. But it may turn out that the employed provisions are "overinclusive" and lead to overreaching.

{81} Employed provisions may be appropriate for protection of the licensor's rights in the second aspect, i.e. for protection of the design, structure and implementation of the software system. Most likely, these will be considered an expression of the idea and will be afforded federal copyright law protection.

{82} However, as for the first aspect, such provisions may lead to overreaching. The prohibition of the copying and reverse-engineering is intended to apply to the components of the lowest levels of software hierarchy interacting with the hardware (as well as to all other components). This may effectively protect the "ideas," but not the "expressions" of ideas. It may also make it impossible for the licensee to obtain the information necessary to provide compatibility between the software and hardware components. For example, if the licensee decides to develop its own programs to be used on the hardware system which the licensee had already acquired, such protection may not be available for the licensor, according to the *Sega* and *Atari* courts.[131] Under the *Vault* decision, if such restrictive provisions are included in the license agreement and the licensor later brings an action for breach of a license agreement, the court may hold that the restrictive provisions touch upon the area of federal copyright law and constitute overreaching.

{83} In addition, the impermissible restriction on copying and reverse-engineering of the program components responsible for the compatibility of the software system with the hardware and communication devices may be interpreted as an intentional restriction of competitors' access to the market. This may not only subject the licensor to accusations of antitrust violations, but also may allow the licensee to successfully raise the copyright misuse defense, according to the *Lasercomb* court.[132] Additionally, this may also lead to an invalidation of the license agreement because of overreaching restrictions on copying and reverse-engineering.

{84} Under these circumstances, the licensor may argue that federal copyright law preemption should not apply to its claim under state trade secret law or state contract law, which is valid under the "equivalence of rights" doctrine.[133] The licensor's argument may be premised upon the contention that its claim includes an extra element - a breach of confidentiality provision under the license agreement - which is not required for bringing an action for copyright infringement. However, the licensor may face significant difficulties defending such a position.

{85} In order to be treated as confidential/proprietary, the information should be disclosed to the licensee by the licensor or by a third party with the licensor's consent. In the case where the licensee attempts to reverse-engineer the program, the software system was obviously delivered in object code, which is readable by the computer, but is generally unreadable by humans. The very purpose of reverse-engineering is to translate the program from object code into source code, which makes further analysis, adaptation and use of the program possible for the licensee, without having to gain permission from the licensor. The licensee may therefore have a valid argument that software in object code is not confidential/proprietary information because it was not disclosed by the licensor. In fact, the licensee may argue that the very prohibition of the reverse engineering in the license agreement is irreconcilable with the contention that the program was disclosed to the licensee. One does not have to reconstruct information if it is already disclosed, and the reverse-engineering constitutes the process of such reconstruction.

{86} Therefore, if the software in object code was not disclosed to the licensee by the licensor and is not confidential/proprietary information, then no confidential relationship as to this information was established between the parties. The copying and reverse engineering of the object code does not breach the confidentiality provisions of the license agreement.

{87} Hence, the licensor fails to satisfy the "extra element" test. Under the "equivalence of rights" doctrine, state law is preempted by federal copyright law. Accordingly, the licensor most likely has no valid claim under the described circumstances because the fair use defense probably applies.

{88} In order to avoid these types of problems, the licensor has several options.

{89} The licensor may decide to separately define the software components of the lowest logical level which interacts with the hardware and communication equipment. Then he may define the rest of the software

system. As to the software components of the lowest level, the licensor may choose to use the full range of intellectual property protection provisions, including security, non-use and non-disclosure clauses, with the exception of activity-restricting provisions that prohibit reverse-engineering and equivalent procedures. As to the rest of the software system, the prohibition of reverse-engineering in addition to security, non-use, non-disclosure and other protective provisions may be appropriate.

{90} The licensor may also consider providing the source code of the components of the lowest level, or disclosing the information about the interface with the hardware and communication equipment sufficient to enable the licensee to develop his own communication software. This allows the licensor to impose stringent restrictions on the possible use of such information, minimizing the chance of impermissible exploitation. Under these circumstances, the licensee loses justification for copying and reverse-engineering the whole system because of the separation of the whole system into two functionally distinct parts. Additionally, confidential relations are established as to the whole system, including the programs of the lowest level. In other words, this strengthens the licensor's position for bringing an action under state trade secret or contract law where the confidential relations are breached by the licensee. Therefore the licensor obtains an additional element for its claim, and avoids preemption of its state law claim by federal copyright law, under the "equivalence of rights" doctrine.

{91} Another benefit for the licensor in employing such an approach is premised on the fact that it establishes a pro-competitive character for the respective non-compete clause.[134] Thus the possibility of the equitable defense of copyright misuse and invalidation of the license agreement on public policy grounds is eliminated. The discussed approach may be selected when there is an insignificant chance of the disclosed information being unlawfully acquired by the licensor's competitors. However, under certain circumstances such an approach may be unacceptable. Where the practical implementation of "policing" procedures is too burdensome or expensive and the probability of misappropriation of disclosed confidential information by licensor's competitors is too high, the proposed approach may be impermissibly risky. This is true even if it seems to be viable from a purely legal standpoint.

{92} As an alternative approach the licensor may consider the use of a source code escrow package.[135] This package includes a complete copy of the source code and executable code of the licensed program, and a complete copy of design documentation and user documentation. It encompasses complete instructions for compiling and linking each part of the source code into executable code, with precise identifications of the components of the system software necessary to generate executable code from the source code. Such a source code escrow package should be delivered to an escrow agent. Certain events can trigger the release of the source code escrow package from the escrow agent to the licensee temporarily or permanently during the pre-defined maintenance period. Ordinarily, such events may include: (a) insolvency or inability of the licensor to pay its debts as they become due (permanent release); (b) filing by the licensor a petition for protection under the Bankruptcy Code (permanent release); (c) acquiring a business of the licensor by a licensee's competitor (permanent release); or (d) inability or failure of the licensor to cure a breach of certain warranties for a predetermined period of time (in this case, the release is temporary for the period of time necessary to effectuate any reasonable attempt of the licensee to cure the breach).[136] If the licensee obtains the right of access to the source code escrow package from the escrow agent upon the occurrence of one of these pre-defined events, the licensee must properly document the occurrence of such an event. All the materials and information comprising the source code escrow package should be maintained in strict confidence and may be used or disclosed only in accordance with the confidentiality provisions of the license agreement. In case of a temporary release, the licensee must promptly return all released materials to the licensor when the circumstances which caused the release are no longer in effect. It is the licensee's obligation to promptly respond to the licensor's requests concerning the use or contemplated use of the source code escrow package and the information about the licensee's employees having access to the package.

{93} Traditionally, one of the main functions of the source code escrow package is to serve as a protective

tool for the licensee when the licensor faces bankruptcy or insolvency problems, where the licensor does not fulfill its support or maintenance obligations or where breaches of warranties take place. However, such a package may also be effectively employed by the licensor to avoid overreaching when related to the prohibition on reverse-engineering of the program object code.

{94} Under the proposed approach, the parties may agree to the additional mutually acceptable pre-defined circumstances which will enable the licensee to temporarily obtain access to the source code or certain parts of the source code. For example, this would be applicable where the program components interact with hardware, or where the information is sufficient to reconstruct the interface between communication software components and hardware. The licensee's need to obtain the information about the interface with hardware and communication equipment to make certain program modifications with its own resources, or develop its own programs, may be an example of such predetermined events. If the licensee lawfully obtains access to the source code, it must comply with the established procedural and confidentiality requirements under the control of an escrow agent. Furthermore, the licensee has to properly document the occurrence of an appropriate event and must maintain the obtained information in strict confidence. It may be used only in compliance with the non-use obligation clause, the non-disclosure obligation clause and other relevant clauses of the license agreement.

{95} Thus, the licensor will be able to achieve the desired degree of protection without incurring the risk of court-declared invalidation of the license agreement due to overreaching.

V. Conclusion

{96} Overreaching occurs where a license agreement places impermissible restrictions on the licensee's activities relating to the licensed software. Under such circumstances, the license agreement may be invalidated in whole or in part. Common issues where overreaching takes place are prohibitions on reverse-engineering coupled with copying of the program, and non-compete clauses which are anticompetitive.

{97} Often overreaching occurs where the licensor is trying to contractually obtain patent-like protection for licensed software without meeting the rigorous requirements of patent law. It also occurs where the software sought to be protected contains functional elements unprotected under copyright law, and where reverse-engineering is the only method available for the licensee to understand how the licensed software works to achieve compatibility with other components of its computer system.

{98} In decisions that hold license agreements unenforceable because of overreaching, courts rely on the doctrine of federal law preemption, the fair use defense under federal copyright law and the equitable defense of copyright misuse. To resolve the issue of federal copyright law preemption, the courts employ the "equivalence of rights" doctrine and the "extra element" test. Moreover, courts are willing to actively apply public policy considerations aimed at the protection of the market competition from unreasonable restraints and promoting the free flow of useful ideas protected under patent law.

{99} To minimize the risk of overreaching, the licensor may perform an analysis aimed at determining which elements of the program are either protected or unprotected under federal copyright law. The licensor should then apply adequate means of contractual protection to the different elements of the program, in addition to careful drafting (using a severability clause) and refraining from anticompetitve provisions.

{100} The proposed approach may provide the licensee with a "viable alternative" to reverse-engineering where the licensee needs access to unprotected elements of the licensed software. In many cases, even though the licensee does not engage in the unauthorized copying or reverse-engineering of the licensed software, the licensee may still run the risk of being sued for copyright infringement and breach of license agreement. As a

result, in such cases it will eliminate the need for a fair use defense, and it may even make such a defense unavailable. At the same time this allows the licensor to achieve the desired level of protection without facing the risk of the court invalidating the license agreement because of overreaching. Both parties therefore may benefit from the increased certainty and predictability of their relationships under the software license agreements.

Download this article

In addition to browsing this article online, you may download this article in either ASCII or PGP formats. If you would like to see other file formats supported, please let us know by sending us e-mail at jolt@richmond.edu.

See the related readings

Journal staff members have compiled a list of hypertext links of information contained on the Internet that may be of interest to you.

Footnotes

[**]**NOTE:** All endnote citations in this article follow the conventions appropriate to the edition of THE BLUEBOOK: A UNIFORM SYSTEM OF CITATION that was in effect at the time of publication. When citing to this article, please use the format required by the Seventeenth Edition of THE BLUEBOOK, provided below for your convenience.

Michael Liberman, Comment, *Overreaching Provisions in Software License Agreements*, 1 RICH. J.L. & TECH. 4 (1995) http://www.richmond.edu/jolt/v1i1/liberman.html.

[1] *See generally*, MICHAEL D. SCOTT, SCOTT ON COMPUTER LAW § 12.08 (a)-(c) (2d ed. 1991).

[2] THE AMERICAN HERITAGE DICTIONARY 1292 (3d ed. 1992).

[3] In this paper the pronoun "it" will be used in reference to the terms "licensor" and "licensee" unless the context otherwise dictates. The impersonal pronoun is chosen because, in the context of software licensing, the licensor and the licensee are ordinarily organizations and entities as opposed to individuals.

[4] *See infra* notes 8-14, 19-36 and accompanying text.

[5] See e.*g*., Sears, Roebuck & Co. v. Stiffel Co., 376 U.S. 225 (1964) (patent and copyright law); Compco Corp. v. Day-Brite Lighting, Inc., 376 U.S. 234 (1964) (patent and copyright law); Bonito Boats, Inc. v. Thunder Craft Boats, Inc., 489 U.S. 141 (1989) (trade secret and contract law).

[6] *Bonito*, 489 U.S. at 152. The doctrine of federal law preemption was later codified in the Copyright Act. 17 U.S.C. § 301 (1988).

[7] *Sears*, 376 U.S. at 229 (citing Sola Elec. Co. v. Jefferson Elec. Co., 317 U.S. 173, 176 (1942)).

[8] U.S. CONST. Art. I, § 8, cl. 8.

[9] 35 U.S.C. § 154 (1988).

[10] 35 U.S.C. § 101 (1988).

[11] 35 U.S.C. §§ 101-103 (1988).

[12] 17 U.S.C. § 102(a) (1988 & Supp. IV 1992).

[13] 17 U.S.C. § 102(b) (1988 & Supp. IV 1992).

[14] 17 U.S.C. § 301(a) (1988).

[15] 17 U.S.C. § 101 (1988).

[16] *See* Apple Computer v. Franklin Computer Corp., 714 F.2d 1240, 1249 (3d Cir. 1983). *See infra*, note 44 For a discussion of the terms "source code" and "object code."

[17] *See e.g.*, Gates Rubber Co. v. Bando Chem. Indus., 9 F.3d 823 (10th Cir. 1993) (applying a modified abstraction-filtration-comparison approach); Computer Assocs. Int'l. v. Altai, Inc., 982 F.2d 693 (2d Cir. 1992) (applying an abstraction-filtration-comparison approach); Whelan Assocs. v. Jaslow Dental Lab., 797 F.2d 1222 (3d Cir. 1986) (protecting the structure, sequence and organization of program), *cert. denied*, 479 U.S. 1031 (1987); Lotus Dev. Corp. v. Borland Int'l, 799 F. Supp. 203 (D. Mass. 1992), *rev'd*, 63 U.S.L.W. 2565 (1st Cir. 1995) (applying a levels of formulation of the idea approach).

[18] 17 U.S.C. § 106 (1988).

[19] 55 AM. JUR. 2D *Monopolies* § 710.3 (1994).

[20] 12 ROGER M. MILGRIM, MILGRIM ON TRADESECRETS § 1.01(2)(b) (1994). Thirty four states have adopted the Uniform Trades Secrets Act. For a complete list see *id*.

[21] VA. CODE ANN. § 59.1-336 (Michie 1992 Replacement Vol.).

[22] RESTATEMENT OF TORTS § 757 cmt. b (1939).

[23] Dionne v. Southeast Foam Converting & Packaging, 397 S.E.2d 110, 113 n.2 (Va. 1990).

[24] 2 STEVEN Z. SZCZEPANSKI, ECKSTROM'S LICENSING IN FOREIGN AND DOMESTIC OPERATIONS § 12.02[8], at 12-31 (1995).

[25] Kewanee Oil Co. v. Bicron Corp., 416 U.S. 470, 476 (1974).

[26] *See* Jostens, Inc. v. National Computer Systems, 318 N.W.2d 691 (Minn. 1982).

[27] *See* 2 SZCZEPANSKI, *supra* note 24.

[28] 17 U.S.C. § 109(b),(e) (Supp. V 1993) (amending 17 U.S.C. § 109 (1988)).

[29] 17 U.S.C. § 109(b).

[30] 17 U.S.C. § 109(a).

[31] RESTATEMENT (SECOND) OF CONTRACTS § 178 (1979).

[32] *Id*. § 186. [33] Edmund W. Kitch, *Intellectual Property and the Common Law*, 78 VA. L. REV. 293, 303 (1992).

[34] *Id*. at 301.

[35] *Id*. at 303-04.

[36] 847 F.2d 255 (5th Cir. 1988); *see infra* discussion in Section III.

[37] Thomas M. S. Hemnes, *Restraints on Alienation, Equitable Servitudes, and the Feudal Nature of Computer Software Licensing*, 71 DENV. U. L. REV. 577, 581 (1994).

[38] Ronald E. Myrick & Penelope S. Wilson, *Licensing Rights to Software*, *in* TECHNOLOGY LICENSING AND LITIGATION 1993, at 467 (PLI Patents, Copyrights, Trademarks, and Literary Property Course Handbook Series No. 354, 1993).

[39] 977 F.2d 1510 (9th Cir. 1992).

[40] 975 F.2d 832 (Fed. Cir. 1992).

[41] Box-top, or shrink-wrap, licenses contain pre-printed terms of the agreement between the software publisher and the customer which the customer is supposed to accept by opening the package with a software product. Ordinarily, such terms can state that the customer has not acquired title to the software but merely obtained a non-transferable personal license to use the software. For a detailed discussion of box-top licenses see David A. Einhorn, *Box-top Licenses and the Battle-of-the-forms*, 5 SOFTWARE L.J. 401, 418 (1992).

[42] 847 F.2d 255 (5th Cir. 1988).

[43] 911 F.2d 970 (4th Cir. 1990).

[44] Using programming languages, computer programmers develop software in the form of source code, which is readable by humans. The software is then translated into the object code, in which the program is represented by the combination of 1s and 0s. Then the software is transformed into the executable code, which can be run on the computer. As a practical matter, in order to understand the program and to intelligently and effectively modify or improve it, programmers need access to the source code. Reverse-engineering, decompiling and disassembling are functionally similar procedures of reconstructing the text of the program by converting software from the forms suitable to usage by the computer back into the source code.

[45] *Vault*, 847 F.2d at 257 & n.2.

[46] *Id*. at 258.

[47] LA. REV. STAT. ANN. §§ 51:1961-1966 (West 1987 & Supp. 1995).

[48] *Id*. §§ 51:1431-1439.

[49] 17 U.S.C. § 117 (1988 & Supp. V 1993).

[50] *Vault*, 847 F.2d at 261 (referring to 17 U.S.C. § 106(1) (1988 & Supp. V 1993)).

[51] 17 U.S.C. § 106(2) (1988 & Supp. V 1993).

[52] *Vault*, 847 F.2d at 267 (citing Litchfield v. Spielberg, 736 F.2d 1352, 1357 (9th Cir. 1984)).

[53] *Id*. at 267-68.

[54] Whelan Assocs. v. Jaslow Dental Lab., 797 F.2d 1222, 1245 (3d Cir. 1986).

[55] *Vault*, 847 F.2d at 268.

[56] LA. REV. STAT. ANN. § 51:1964 (West 1987 & Supp. 1995).

[57] *Vault*, 847 F.2d at 269 (comparing La. Rev. Stat. Ann. §§ 51:1961-1966 (West 1987 & Supp. 1995) with 17 U.S.C. §§ 101-810 (1988 & Supp. V 1993).

[58] Vault Corp. v. Quaid Software Ltd., 655 F. Supp. 750, 761 (E.D. La. 1987), *aff'd* 847 F.2d 255 (5th Cir. 1988).

[59] *Vault*, 847 F.2d at 269-70 (citing Sears, Roebuck & Co. v. Stiffel Co., 376 U.S. 225, 229 (1964).

[60] *Id*. at 270.

[61] 911 F.2d 970 (4th Cir. 1990).

[62] *Id*. at 973.

[63] *Id*.

[64] *Id*. at 971.

[65] *Id*. at 972.

[66] Lasercomb America v. Holiday Steel Rule Die Corp., 656 F. Supp. 612, 616 (M.D.N.C.), *appeal dismissed*, 829 F.2d 36 (4th Cir. 1987).

[67] *Reynolds*, 911 F.2d at 972.

[68] *Id*. at 978.

[69] *Id*. at 979.

[70] *Id*. at 978.

[71] *Id*. at 980.

[72] 977 F.2d 1510 (9th Cir. 1992).

[73] *Id*. at 1514-15.

[74] *Id*. at 1518.

[75] 17 U.S.C. § 107 (1988 & Supp. V 1993).

[76] *Sega*, 977 F.2d at 1522.

[77] *Id*. at 1523.

[78] 17 U.S.C. § 102(b) (1988 & Supp. V 1993).

[79] *See Sega*, 977 F.2d at 1524.

[80] *Id*. at 1526.

[81] *Id*.

[82] *Id*. at 1526-27.

[83] *Id*. at 1523.

[84] *Id*.

[85] *Id*. at 1523-24.

[86] *Id*. at 1527-28.

[87] 975 F.2d 832 (Fed. Cir. 1992).

[88] *Id*. at 836, 842.

[89] *Id*. at 836.

[90] Atari Games v. Nintendo of Am., Nos. 88-4805, 89-0027, 89-0824, 1991 WL 57304, at *6 (N.D.Cal. Apr. 11, 1991) *aff'd*, 975 F.2d 832 (Fed. Cir. 1992).

[91] *Atari*, 975 F.2d at 844.

[92] *Id*. at 843.

[93] *Id*.

[94] *Id*.

[95] Vault Corp. v. Quaid Software, 847 F.2d 255 (5th Cir. 1988).

[96] 719 F. Supp. 1006 (D. Kan. 1989).

[97] *Id*. at 1011.

[98] *See* Computer Assocs. v. Altai, Inc., 982 F.2d 693 (2d Cir. 1992).

[99] *See* Page M. Kaufman, Note, *The Enforceability of State "Shrink-Wrap" License Statutes in Light of* Vault Corp. v. Quaid Software, Ltd., 74 CORNELL L. REV. 222, 230 (1988).

[100] *Foresight*, 719 F.Supp. at 1010 (citing Vault Corp. v. Quaid Software Ltd., 847 F.2d 255, 270 (5th Cir. 1988)).

[101] Lasercomb America v. Reynolds, 911 F.2d 970, 978 (4th Cir. 1990).

[102] 17 U.S.C. § 107. *See also* Sega Enters. v. Accolade, Inc., 977 F.2d 1510 (9th Cir. 1992); Atari Games Corp. v. Nintendo of America, 975 F.2d 832 (Fed. Cir. 1992).

[103] *Lasercomb*, 911 F.2d at 978.

[104] *Id*.

[105] Vault Corp. v. Quaid Software, 847 F.2d 255, 269 (5th Cir. 1988).

[106] 939 F.2d 91 (3rd Cir. 1991).

[107] *Id*. at 98.

[108] Sega Enters. v. Accolade, Inc., 977 F.2d 1510, 1527 (9th Cir. 1992).

[109] *Vault*, 847 F.2d at 269.

[110] Lotus Dev. Corp. v. Borland Int'l, 799 F. Supp. 203 (D. Mass. 1992), *rev'd*, 63 U.S.L.W. 2565 (1st Cir. 1995).

[111] Computer Assocs. Int'l. v. Altai, Inc., 982 F.2d 693 (2d Cir. 1992).

[112] Gates Rubber Co. v. Bando Chem. Indus., 9 F.3d 823 (10th Cir. 1993).

[113] *Vault*, 847 F.2d at 261. It is legal for the licensee to perform or authorize, without the licensor's permission, the copying or adaptation of the program if the resulted copy or adaptation is "created as an essential step in the utilization of the program in conjunction with a machine." 17 U.S.C. § 117(1).

[114] *Id*.

[115] *See generally* D.C. Toedt III, *Model Software License Provisions, in* THE LAW AND BUSINESS OF COMPUTER SOFTWARE 18-1 to -149 (D.C. Toedt III ed., 1994).

[116] *Id*. at 18-17.

[117] *Id*.

[118] *Id*.

[119] *Id*.

[120] *Id*. at 18-18.

[121] *Id*.

[122] *Id*.

[123] *Id*.

[124] 2 DON A. ALLEN & LARRY J. DAVIS, ALLEN AND DAVIS ON COMPUTER CONTRACTING 904-10 (2d ed. 1992).

[125] *See* Toedt, *supra* note 114. A security conditions clause sets the standard for the protection of

confidential/proprietary information which should be maintained by the licensee (usually on the same level as the one for protection of licensee's own confidential information). A non-use obligation clause prohibits the licensee from using the confidential/proprietary information during certain a limited time upon termination of the license (except as for the benefit of the licensor or with a licensor's prior approval). A non-disclosure obligation clause contains standard provisions prohibiting disclosure of the confidential/proprietary information to any third party without prior consent of the licensor, restricting the personnel of the licensee which shall have access to the proprietary information and determining the procedures to which such personnel shall comply in order to have confidentiality of the disclosed proprietary information being enforced.

[126] Myrick and Wilson, *supra* note 38.

[127] 17 U.S.C. § 107.

[128] 17 U.S.C. § 117.

[129] *See* Vault Corp. v. Quaid Software, 847 F.2d 255 (5th Cir. 1988); Lasercomb America v. Reynolds, 911 F.2d 970 (4th Cir. 1990).

[130] *Vault*, 847 F.2d at 267-68.

[131] Sega Enters. v. Accolade, Inc., 977 F.2d 1510, 1514 (9th Cir. 1992); Atari Games Corp. v. Nintendo of America, 975 F.2d 832, 844 (Fed. Cir. 1992).

[132] Lasercomb America v. Reynolds, 911 F.2d 970, 974 (4th Cir. 1990).

[133] 17 U.S.C § 301(b)(3).

[134] Philip Abromats, *Copyright Misuse and Anticompetitive Software Licensing Restrictions*: Lasercomb America, Inc. v. Reynolds, 52 U. PITT. L. REV. 629, 653 (1991) ("where noncompete provisions might actually enhance competition by providing protection to program developers that would enable them to write software that would otherwise go undeveloped.").

[135] *See* Toedt, *supra* note 114, § 261.1.

[136] *Id*. § 261.6.
